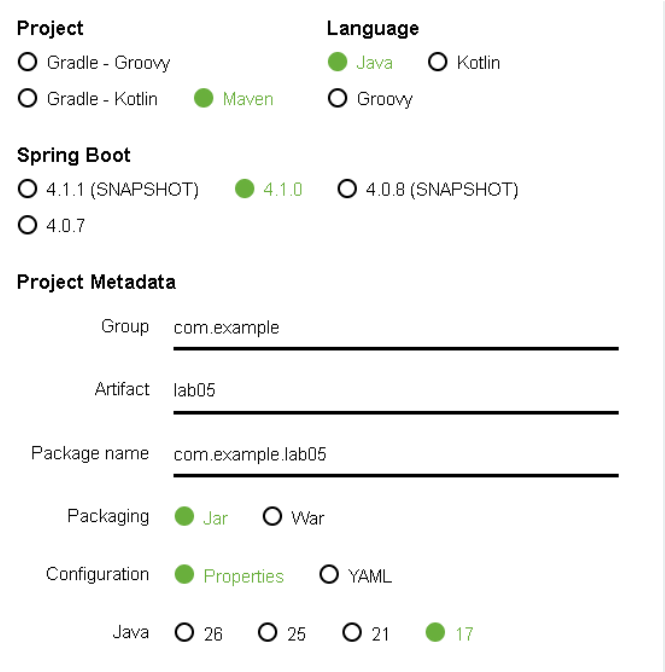
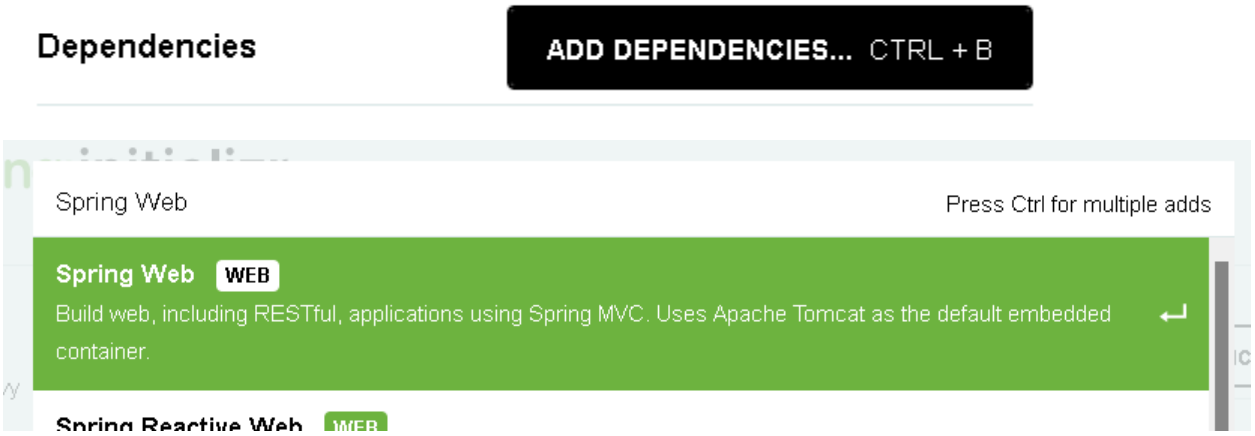
673380300-2 เมธัส มณีวิจิตร sec1

เปิดเว็บ start.spring.io แล้วเลือก option ตามภาพ




เพิ่ม Dependency **Spring Web**




กด generate

Project Structure

```
src/main/java/com/example/coffeemenu/
├── CoffeeMenuApplication.java
├── model/
│   └── Coffee.java
├── service/
│   └── CoffeeService.java
└── controller/
    └── CoffeeController.java
```

## 1. Model Layer (Coffee.java)

```java
package com.example.coffeemenu.model;

public class Coffee {
    private Long id;
    private String name;
    private double price;

    // Default Constructor (จำเป็นสำหรับ JSON Deserialization)
    public Coffee() {
    }

    public Coffee(Long id, String name, double price) {
        this.id = id;
        this.name = name;
        this.price = price;
    }

    // Getters and Setters
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }
}
```

## 2. Service Layer (CoffeeService.java)

```java
package com.example.coffeemenu.service;

import com.example.coffeemenu.model.Coffee;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.List;
import java.util.Optional;
import java.util.concurrent.atomic.AtomicLong;

@Service
public class CoffeeService {

    private final List<Coffee> coffeeList = new ArrayList<>();
    private final AtomicLong idCounter = new AtomicLong(0);

    public CoffeeService() {
        // ข้อมูลตัวอย่างตั้งต้น 2 รายการ เมื่อเริ่มรันโปรแกรม
        addCoffee(new Coffee(null, "Espresso", 45.0));
        addCoffee(new Coffee(null, "Latte", 55.0));
    }

    // 1. ดูเมนูทั้งหมด
    public List<Coffee> getAllCoffees() {
        return coffeeList;
    }

    // 2. ดูเมนูตาม ID
    public Optional<Coffee> getCoffeeById(Long id) {
        return coffeeList.stream()
                .filter(c -> c.getId().equals(id))
                .findFirst();
    }

    // 3. เพิ่มเมนูใหม่ (Gen ID อัตโนมัติ)
    public Coffee addCoffee(Coffee coffee) {
        long newId = idCounter.incrementAndGet();
        coffee.setId(newId);
        coffeeList.add(coffee);
        return coffee;
    }

    // 4. แก้ไขเมนูเดิมตาม ID
    public Optional<Coffee> updateCoffee(Long id, Coffee updatedCoffee) {
        return getCoffeeById(id).map(existingCoffee -> {
            existingCoffee.setName(updatedCoffee.getName());
            existingCoffee.setPrice(updatedCoffee.getPrice());
            return existingCoffee;
        });
    }

    // 5. ลบเมนูตาม ID
    public boolean deleteCoffee(Long id) {
        return coffeeList.removeIf(coffee -> coffee.getId().equals(id));
    }

    // (Bonus) ค้นหาเมนูตามชื่อ
    public List<Coffee> searchCoffeesByName(String name) {
        return coffeeList.stream()
                .filter(coffee -> coffee.getName().toLowerCase().contains(name.toLowerCase()))
                .toList();
    }
}
```

## 3. Controller Layer (CoffeeController.java)

```java
package com.example.coffeemenu.controller;

import com.example.coffeemenu.model.Coffee;
import com.example.coffeemenu.service.CoffeeService;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/coffees")
public class CoffeeController {

    private final CoffeeService coffeeService;

    // Dependency Injection ผ่าน Constructor
    public CoffeeController(CoffeeService coffeeService) {
        this.coffeeService = coffeeService;
    }

    // 1. GET /coffees - ดูเมนูทั้งหมด
    @GetMapping
    public ResponseEntity<List<Coffee>> getAllCoffees() {
        return ResponseEntity.ok(coffeeService.getAllCoffees());
    }

    // 2. GET /coffees/{id} - ดูเมนู 1 รายการตาม ID (คืน 404 ถ้าไม่พบ)
    @GetMapping("/{id}")
    public ResponseEntity<Coffee> getCoffeeById(@PathVariable Long id) {
        return coffeeService.getCoffeeById(id)
                .map(ResponseEntity::ok)
                .orElseGet(() -> ResponseEntity.status(HttpStatus.NOT_FOUND).build());
    }

    // 3. POST /coffees - เพิ่มเมนูใหม่
    @PostMapping
    public ResponseEntity<Coffee> addCoffee(@RequestBody Coffee coffee) {
        Coffee createdCoffee = coffeeService.addCoffee(coffee);
        return ResponseEntity.status(HttpStatus.CREATED).body(createdCoffee);
    }

    // 4. PUT /coffees/{id} - แก้ไขเมนูเดิมตาม ID (คืน 404 ถ้าไม่พบ)
    @PutMapping("/{id}")
    public ResponseEntity<Coffee> updateCoffee(@PathVariable Long id, @RequestBody Coffee coffee) {
        return coffeeService.updateCoffee(id, coffee)
                .map(ResponseEntity::ok)
                .orElseGet(() -> ResponseEntity.status(HttpStatus.NOT_FOUND).build());
    }

    // 5. DELETE /coffees/{id} - ลบเมนูตาม ID (คืน 404 ถ้าไม่พบ)
    @DeleteMapping("/{id}")
    public ResponseEntity<Void> deleteCoffee(@PathVariable Long id) {
        boolean deleted = coffeeService.deleteCoffee(id);
        if (deleted) {
            return ResponseEntity.noContent().build(); // 204 No Content
        } else {
            return ResponseEntity.status(HttpStatus.NOT_FOUND).build();
        }
    }

    // (Bonus) GET /coffees/search?name=... - ค้นหาตามชื่อ
    @GetMapping("/search")
    public ResponseEntity<List<Coffee>> searchCoffees(@RequestParam String name) {
        return ResponseEntity.ok(coffeeService.searchCoffeesByName(name));
    }
}
```

4. Main Application (CoffeeMenuApplication.java)

```java
package com.example.coffeemenu;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class CoffeeMenuApplication {

    public static void main(String[] args) {
        SpringApplication.run(CoffeeMenuApplication.class, args);
    }

}
```

แก้ pom.xml

```xml
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>
```

```xml
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
      <version>4.1.0</version>
      <configuration>
        <mainClass>com.example.coffeemenu.CoffeeMenuApplication</mainClass>
      </configuration>
      <executions>
        <execution>
          <goals>
            <goal>repackage</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
```

รัน mvn compile ใน terminal

```
PS C:\Users\Admin\Desktop\lab05> mvn clean compile
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------< com.example:lab05 >-------------------------
[INFO] Building  0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- clean:3.5.0:clean (default-clean) @ lab05 ---
[INFO] Deleting C:\Users\Admin\Desktop\lab05\target
[INFO]
[INFO] --- resources:3.5.0:resources (default-resources) @ lab05 ---
[INFO] Copying 1 resource from src\main\resources to target\classes
[INFO] Copying 0 resource from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.15.0:compile (default-compile) @ lab05 ---
[INFO] Recompiling the module because of changed source code.
[INFO] Compiling 5 source files with javac [debug parameters release 17] to target\classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  2.526 s
[INFO] Finished at: 2026-08-01T19:54:41+07:00
[INFO] ------------------------------------------------------------------------
PS C:\Users\Admin\Desktop\lab05> mvn exec:java -Dexec.mainClass="com.example.coffeemenu"
```

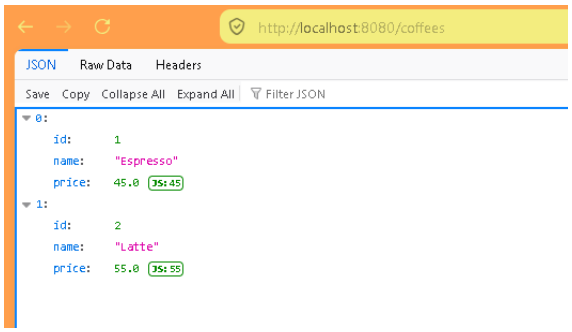
## รัน mvn spring-boot:run ใน terminal

```
PS C:\Users\Admin\Desktop\lab05> mvn spring-boot:run
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------< com.example:lab05 >-------------------------
[INFO] Building  0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] >>> spring-boot:4.1.0:run (default-cli) > test-compile @ lab05 >>>
[INFO]
[INFO] --- resources:3.5.0:resources (default-resources) @ lab05 ---
[INFO] Copying 1 resource from src\main\resources to target\classes
[INFO] Copying 0 resource from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.15.0:compile (default-compile) @ lab05 ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- resources:3.5.0:testResources (default-testResources) @ lab05 ---
[INFO] skip non existing resourceDirectory C:\Users\Admin\Desktop\lab05\src\test\resources
[INFO]
[INFO] --- compiler:3.15.0:testCompile (default-testCompile) @ lab05 ---
[INFO] Recompiling the module because of changed source code.
[INFO] Compiling 1 source file with javac [debug parameters release 17] to target\test-classes
[INFO]
[INFO] <<< spring-boot:4.1.0:run (default-cli) < test-compile @ lab05 <<<
[INFO]
[INFO]
[INFO] --- spring-boot:4.1.0:run (default-cli) @ lab05 ---
[INFO] Attaching agents: []

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v4.1.0)

2026-08-01T19:55:19.762+07:00  INFO 5872 --- [lab05] [           main] c.e.coffeemenu.CoffeeMenuApplication     : Starting CoffeeMenuApplication using Java 17.
0.12 with PID 5872 (C:\Users\Admin\Desktop\lab05\target\classes started by Beamy in C:\Users\Admin\Desktop\lab05)
2026-08-01T19:55:19.766+07:00  INFO 5872 --- [lab05] [           main] c.e.coffeemenu.CoffeeMenuApplication     : No active profile set, falling back to 1 defa
ult profile: "default"
2026-08-01T19:55:20.395+07:00  INFO 5872 --- [lab05] [           main] o.s.boot.tomcat.TomcatWebServer          : Tomcat initialized with port 8080 (http)
2026-08-01T19:55:20.405+07:00  INFO 5872 --- [lab05] [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2026-08-01T19:55:20.405+07:00  INFO 5872 --- [lab05] [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/11.0.
22]
2026-08-01T19:55:20.448+07:00  INFO 5872 --- [lab05] [           main] b.w.c.s.WebApplicationContextInitializer : Root WebApplicationContext: initialization co
mpleted in 629 ms
2026-08-01T19:55:20.693+07:00  INFO 5872 --- [lab05] [           main] o.s.boot.tomcat.TomcatWebServer          : Tomcat started on port 8080 (http) with conte
xt path '/'
2026-08-01T19:55:20.700+07:00  INFO 5872 --- [lab05] [           main] c.e.coffeemenu.CoffeeMenuApplication     : Started CoffeeMenuApplication in 1.329 second
s (process running for 1.661)
```

ตัวอย่างการทดสอบ API

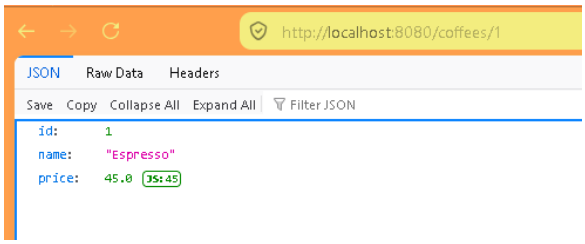
1. ดูรายการเมนูกาแฟทั้งหมด (GET All)

```
C:\Users\Admin\Desktop\lab05>curl http://localhost:8080/coffees
[{"id":1,"name":"Espresso","price":45.0},{"id":2,"name":"Latte","price":50.0}]
C:\Users\Admin\Desktop\lab05>
```




2. ดูรายการเมนูกาแฟตาม ID (GET by ID)

```
C:\Users\Admin\Desktop\lab05>curl http://localhost:8080/coffees/1
{"id":1,"name":"Espresso","price":45.0}
```

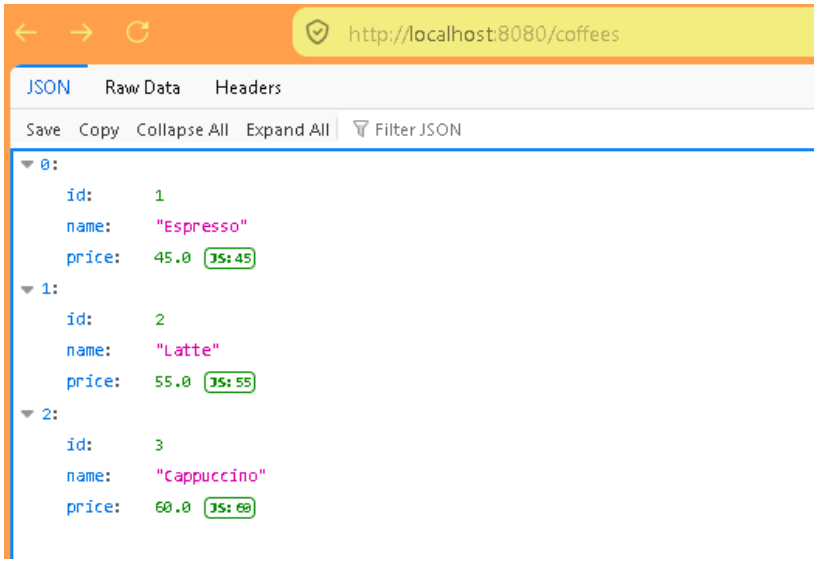



3. เพิ่มเมนูกาแฟใหม่ (POST)

เปิด command prompt และใส่ command

```
curl -X POST http://localhost:8080/coffees ^-H "Content-Type: application/json" ^-d "{\"name\": \"Cappuccino\", \"price\": 60.0}"
```

```
C:\Users\Admin\Desktop\lab05>curl -X POST http://localhost:8080/coffees ^-H "Content-Type: application/json" ^-d "{\"name\": \"Cappuccino\", \
"price\": 60.0}"
{"id":3,"name":"Cappuccino","price":60.0}
C:\Users\Admin\Desktop\lab05>
```



4. แก้ไขเมนูกาแฟตาม ID (PUT)
เปิด command prompt และใส่ command

```
curl -X PUT http://localhost:8080/coffees/2 ^-H "Content-Type: application/json" ^-d "{\"name\": \"Latte\", \"price\": 50.0}"
```

```
C:\Users\Admin\Desktop\lab05>curl -X PUT http://localhost:8080/coffees/2 ^-H "Content-Type: application/json" ^-d "{\"name\": \"Latte\", \"price\": 50.0}"
{"id":2,"name":"Latte","price":50.0}
C:\Users\Admin\Desktop\lab05>
```

```
id:      2
name:    "Latte"
price:   50.0  JS: 50
```

5. ลบเมนูกาแฟตาม ID (DELETE)
เปิด command prompt และใส่ command

```
curl -X DELETE http://localhost:8080/coffees/3
```

```
C:\Users\Admin\Desktop\lab05>curl -X DELETE http://localhost:8080/coffees/3

C:\Users\Admin\Desktop\lab05>
```

http://localhost:8080/coffees

JSON  Raw Data  Headers

Save  Copy  Collapse All  Expand All  Filter JSON

```
0:
    id:      1
    name:    "Espresso"
    price:   45.0  JS: 45
1:
    id:      2
    name:    "Latte"
    price:   50.0  JS: 50
```

เพิ่ม endpoint ค้นหาตามชื่อ GET /coffees/search?name=...

```
C:\Users\Admin\Desktop\lab05>curl http://localhost:8080/coffees/search?name=latte
[{"id":2,"name":"Latte","price":50.0}]
C:\Users\Admin\Desktop\lab05>
```

คืน 404 Not Found เมื่อหา id ไม่เจอ

http://localhost:8080/coffees/5

## Looks like there's a problem with this site

The server at **localhost:8080** sent back an error: 404 Not Found

### What can you do about it?

- Check to make sure you've typed the website address correctly.
- The site could be temporarily unavailable or too busy. Try again in a few moments.

Try Again

ตอบส่วน Discussion

1. HTTP method แต่ละตัว (GET/POST/PUT/DELETE) ต่างกันอย่างไร ยกตัวอย่างจากโปรเจกต์ตัวเอง
**GET:** ใช้สำหรับ **อ่าน/ดึงข้อมูล** โดยไม่มีการแก้ไขข้อมูลในระบบ เช่น `GET /coffees` เพื่อเรียกดูรายการเมนูกาแฟทั้งหมด
**POST:** ใช้สำหรับ **สร้าง/เพิ่มข้อมูลใหม่** เช่น `POST /coffees` พร้อมส่ง JSON เพื่อบันทึกเมนูกาแฟใหม่ลงในระบบ
**PUT:** ใช้สำหรับ **แก้ไข/อัปเดตข้อมูลที่มีอยู่แล้ว** เช่น `PUT /coffees/2` เพื่อแก้ไขชื่อหรือราคาของเมนูกาแฟ ID 2
**DELETE:** ใช้สำหรับ **ลบข้อมูล** ออกจากระบบ เช่น `DELETE /coffees/3` เพื่อลบเมนูกาแฟ ID 3 ออกจาก `List`

2. ทำไมต้องแยก Controller กับ Service ออกจากกัน มีข้อดีอย่างไรถ้าโปรแกรมโตขึ้น

**เหตุผล:** เพื่อทำตามหลัก **Single Responsibility Principle (SRP)** ให้ Controller มีหน้าที่รับ-ส่ง HTTP Request/Response เท่านั้น แล้วยกการคำนวณและจัดการข้อมูล (Business Logic) ให้เป็นหน้าที่ของ Service

**ข้อดีเมื่อโปรแกรมโตขึ้น:** ทำให้โค้ดเป็นระเบียบ อ่านง่าย ซ่อมบำรุงสะดวก สามารถนำ Service ไป Re-use กับ Controller อื่นๆ ได้ (เช่น Web API หรือ Mobile API) และเขียน **Unit Test** ทดสอบ Logic ได้ง่ายโดยไม่ต้องจำลอง HTTP Request

3. ข้อมูลที่เก็บไว้ใน `List` ใน memory หายไปตอนไหน และถ้าอยากให้ไม่หายควรทำอย่างไร
**เมื่อไหร่ที่หาย:** ข้อมูลจะหายไปทันทีเมื่อ **โปรแกรมหยุดทำงาน (Stop/Restart Application)** หรือเมื่อ Server ดับ/ปิดเครื่องลง เพราะข้อมูลอยู่ใน RAM (Memory)
**แนวคิดการแก้ไข:** ต้องนำข้อมูลไปจัดเก็บไว้ในหน่วยความจำถาวร (**Persistent Storage**) เช่น การต่อกับ **Database** (เช่น MySQL, PostgreSQL) หรือการเขียนบันทึกลง **File System** (เช่น JSON, CSV) เพื่อให้ข้อมูลคงอยู่แม้แอปพลิเคชันจะปิดไป

4. Annotations แต่ละตัวทำหน้าที่อะไร
**`@RestController`:** กำหนดให้คลาสนี้เป็น Controller สำหรับ REST API โดยจะแปลง Return Value ของ Method เป็น JSON Response ส่งกลับไปให้ Client โดยอัตโนมัติ
**`@GetMapping`:** ระบุให้ Method รองรับ HTTP **GET** Request ตาม Path ที่กำหนด
**`@PostMapping`:** ระบุให้ Method รองรับ HTTP **POST** Request ตาม Path ที่กำหนด
**`@PathVariable`:** ใช้ดึงค่าตัวแปรจาก **URL Path** (เช่น ค่า `{id}` ใน `/coffees/{id}`) มาใส่ลงใน Parameter ของ Java Method
**`@RequestBody`:** ใช้แปลงข้อมูล **JSON** ที่ถูกส่งมาใน Request Body ให้กลายเป็น Java Object โดยอัตโนมัติ